OPEN ACCESS

University of BRISTOL

Channegowda, M., Vlachogiannis, T., Nejabati, R., & Simeonidou, D. (2016). Optical flyways for handling elephant flows to improve big data performance in SDN enabled Datacenters. In *Optical Fiber Communication Conference* Optical Society of America (OSA). https://doi.org/10.1364/OFC.2016.W3F.2

Peer reviewed version

Link to published version (if available):
10.1364/OFC.2016.W3F.2

Link to publication record in Explore Bristol Research
PDF-document

University of Bristol - Explore Bristol Research
General rights

This document is made available in accordance with publisher policies. Please cite only the published version using the reference above. Full terms of use are available:
http://www.bristol.ac.uk/red/research-policy/pure/user-guides/ebr-terms/

Optical Flyways for Handling Elephant Flows to improve Big Data Performance in SDN enabled Datacenter's

Mayur Channegowda, Tasos Vlachogiannis, Reza Nejabati and Dimitra Simeonidou
High Performance Networks Group, University of Bristol, Bristol, UK
mayur.channegowda@bristol.ac.uk

Abstract: The effect of different high-volume traffic on big data applications impose stringent requirements on networks. We investigate drawbacks of segregating big data and elephant flows and propose ways to address the problem using optical network.
OCIS codes: (060.4253) **General;** (060.0060) General [Fiber optics and optical communications]

1. Introduction

Big data analysis relies on distributed architecture frameworks such as Hadoop®, Spark™ etc. for managing large datasets of unprecedented volume for business analytics. These frameworks leverage their strength over the network infrastructure that their nodes communicate over, syncing with CPU and I/O resources. Behavior and performance of Hadoop® clusters in datacenters is effected by size of nodes, data size and workload types as well as networking characteristics. Network speed and latency play an important role in Hadoop® job completion times but more importantly they are impacted by the availability and resiliency features, traffic bursting nature and subscription ratio.

There have been several studies [1], [2], pointing the benefits of Software Defined Networking (SDN) for handling network impact on big data workloads. Typically they address the network impact from communication patterns of Hadoop®, which is the most popular big data framework owing to its availability and reliability advantage. These Hadoop based SDN studies analyze communication pattern via network devices or via application awareness. Once the pattern is analyzed then end-to-end flows are setup to optimize the network thereby reducing Hadoop® job completion times. In both cases an SDN controller uses a protocol such as OpenFlow to make intelligent routing decisions, configuring flows on queues or use packet scheduling schemes to improve performance.

Furthermore, owing to the communication patterns of big data, optics have renewed interest in data. [3][4][5] We describe new datacenter architecture with SDN for addressing incast (many-to-one), multicast (one-to-many) and All-to-all cast patterns observed in big data based datacenters. They utilize low-power, high bandwidth circuit switches combined with low-cost passive optical devices (splitters, combiners etc) to handle long lived, high volume elephant traffic flows providing better performance for latency sensitive flows.

Fewer studies address the combined effect of big data and other datacenter traffic flows. Since datacenters run different application traffic like web frontend, VM migration, large data transfers etc., along with Hadoop® it is important to investigate how other flows behave with big data flows. Especially the influence of elephant flows, which may decrease Hadoop's® as well as its own performance.

In this paper we look at this effect of elephant flows on Hadoop® traffic and how it increases the job completion times. We demonstrate how existing methods of sharing packet queues via SDN control plane lead to increase latencies, resource utilization and how it affects all traffic types. Then make a case for packet-optical hybrid datacenter utilizing SDN to identify and segregate elephant flows using optical paths to increase performance of all traffic types.

2. Analysis of big data traffic over a SDN based packet network

Hadoop® internals: In order to understand big data infrastructure we use Hadoop® distributed computing framework which consists of Hadoop® MapReduce and Hadoop® Distributed File System (HDFS). The framework exhibits a master slave paradigm where the master (*NameNode* or *JobTracker*) provides the information for running the job to the client and the slaves (*DataNodes*) are used to execute the job. Each job in *MapReduce* specifies a map function that processes the input data, either provided directly by the client or available on the HDFS, to generate a list of intermediate results. Then a user-defined reduce program is called to merge all intermediate results which are stored in each node's local filesystem. This cycle of merging results and running *MapReduce* function on them is called the shuffling phase and runs iteratively until the final result. Shuffling results many-to-one traffic pattern with high volume data which is easily affected by congested network. HDFS is used alternatingly to store both the input to the map and the output of the reduce phase.

Testbed setup to analyze Hadoop® working along with background high volume datacenter traffic: In order to study Hadoop® traffic combined with other datacenter traffic we set up a small scale Helios [4] based hybrid datacenter architecture setup with SDN control. Our setup consists of 16 servers with 8 identical servers with multiple one Gigabit interfaces used for Hadoop® processing and another set of identical servers equipped with multiple

10/40GbE interfaces used to host virtual machines that generate mice and elephant flow traffic. All servers run Hadoop® 2.2 installed on top of Debian 6 Linux operating system. The servers are organized in 4 racks (4 servers per rack) interconnected by four OpenFlow (OF) enabled Top-of-Rack (ToR) switches. The ToR switches are connected to a packet based aggregate switch and an optical switch (Polatis) with 10/40GbE uplinks as shown in Fig 1A. Another server runs an instance of the OpenDaylight controller and connects to all switches through a management network.

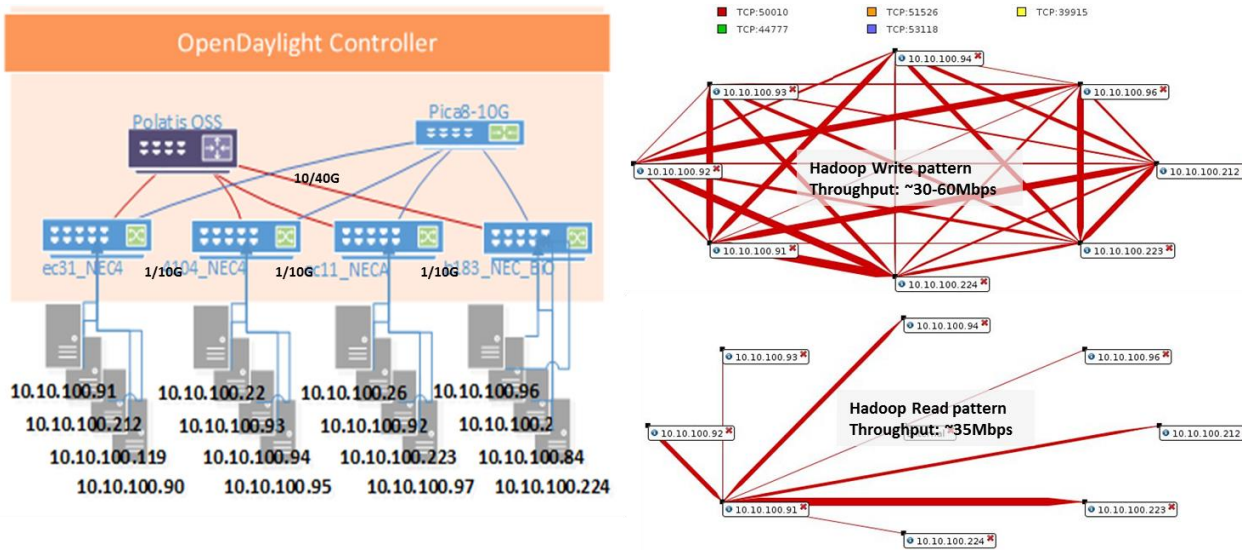

Figure 1 A) Testbed setup with SDN controller, B:Top) mapreduce write flow pattern(All-to-All) with throuhgput around 60Mbps B:Bottom) mapreduce read flow pattern(many-to-one) with throughput around 35Mbps

Using Hadoop's stress tool TestDFSIO we tested Hadoop's write (HDFS) and read (*MapReduce*) performance over the setup. The network pattern is shown in figure Fig.1B (Top and Bottom) which shows a pipelined write pattern (~60Mbps) where each *DataNode* exchange data sequentially and an incast (many to one) type pattern (~35Mbps) for *MapReduce* shuffling process. Next elephant flows i.e. VM migration and large data transfer (~1-9Gbps with duration >10sec) were introduced along with the TestDFSIO flows. Job completion time along with elephant flow latency and loss results were collected (Fig2 C and D) to observe the effect of adding high volume background traffics. The results depict that Hadoop® job completion times increases with addition of background traffic and points to multiple network congestion routes. With the knowledge of the whole network and the application pattern (derived from network or from application directly) an SDN controller can address the aforementioned problem by providing alternative routes for congested paths and using packet priority queuing[1][2] to prioritize *MapReduce* flows. However, these methods have an adverse effect which is detailed in the next section

Drawbacks of using packet queueing for managing different traffic types: Results in Fig.2C&D show the latency & loss for background elephant flows traffic when packet queuing is applied. The problem with elephant flows is that they tend to fill network buffers end-to-end, and this introduces non-trivial queuing delay to any traffic that shares these buffers. So when multiple elephant flows (especially flows >10GbE) share the same queuing buffer they have large latencies with high deviations (shown in Fig.2B) which invariably degrades performance. Furthermore buffering increases memory and CPU resource, shown in Fig.2A, on the packet switches raising power costs.

3. Optical flyways for elephant flows

Handling the noted elephant flows using an optical switch is more beneficial which is best suited for managing multiple bandwidth intensive elephant flows. Moreover the nature of such elephant flows like VM migration, large data transfer exhibit one-to-many pattern which can be further exploited using passive optical splitters and combiners. To demonstrate these benefits we ran the same TestDFSIO experiments but instead of queuing packets on electrical switches an optical switch is used. An SDN controller will identify elephant flows during Hadoop® jobs and then based on the traffic matrix will aggregate multiple elephant flows over the optical paths. This involves 2 steps

Step 1: Identification via monitoring:

- Dynamically detect long lived flows by maintaining per-flow traffic statistics(OF, SFlow etc.) in the network and declaring a flow to be an elephant flow when its byte count exceeds threshold
- Traffic sampling with variable sampling window size (passive-less reliable & active monitoring)

- Derive the stats directly from the application using server socket buffers

We combine the second and third approach where a monitoring application was developed using OpenDaylight REST API to gather flow bytes count via OF protocol with a variable sampling window. Based on the flows and their volume we change the sampling frequency and declare a particular flow to be elephant if its 1) greater than 1/3rd of the overall flows 2) if the duration of the flow is twice the time it takes for the controller to setup a flow (cross connection) on the optical switch (optical switches have higher switching times).

Step 2: Modification via SDN controller which involves

- OpenDaylight controller discovering heterogeneous network topology (packet & optical)
- Monitoring application triggers elephant flow migration requests to ODL
- Controller pushes appropriate QoS based flows to packet & optical switches to handle elephant flows

Once the elephant flows are discovered our application pushes flows to packet switches to enqueue Hadoop flows for higher priority and also pushes flows to optical switches to reroute elephant flows over for higher bandwidth. Fig.2A, C&D show the difference between the approaches. Once the elephant flows are handled through the optical path we can see that CPU & Memory decreases in A&B. In C&D the optical path shows the same job completion times but the latency and loss of the elephant flows drastically reduces improving overall performance.

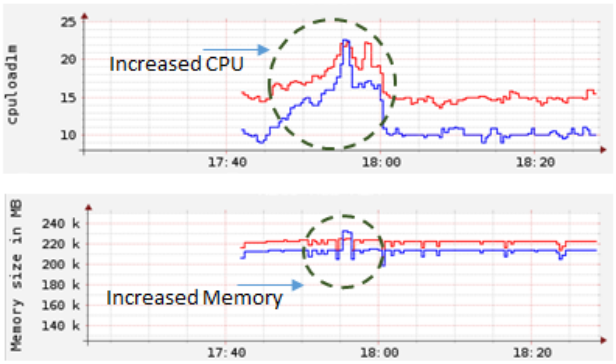

Fig 2A: **ToR and aggregate switch CPU and Memory utilization**

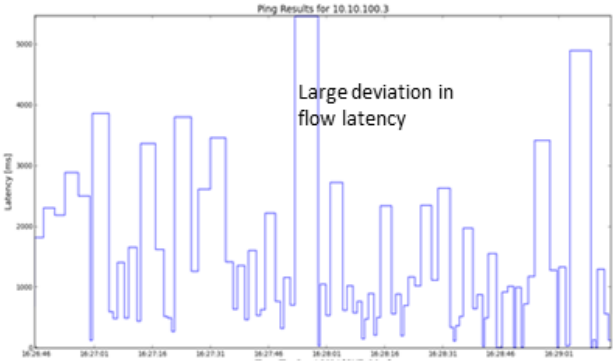

Fig 2B: **Ping latency to measure elephant aggregation at queues**

Fig 2C: **read cycle with and without background elephant flows**

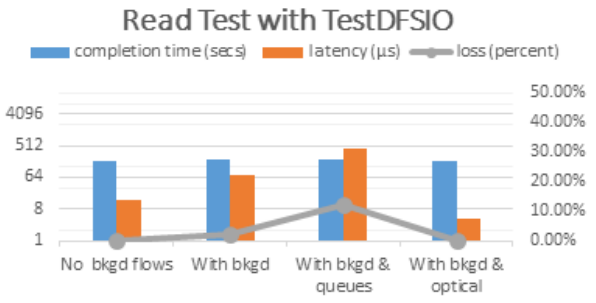

Fig 2D: **read cycle with and without background elephant flows**

Figure 2 A & B measure problems of packet queuing. C & D measure effect of elephant flows on Hadoop

Conclusion

In this work we made an attempt to analyze the network impact of combined elephant and big data flows and showed how optical paths can make a difference. The paper sets a background picture for utilizing optical devices in big data deployments and provides insight on how the research can be further progressed with optical sub systems.

Acknowledgement: This work was supported EPSRC grant EP/L020009/1: TOUCAN project

4. References

[1] Narayan, S.; Bailey, S.; Daga, A., "Hadoop Acceleration in an OpenFlow-Based Cluster," in *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:* , vol., no., pp.535-538, 10-16 Nov. 2012 doi: 10.1109/SC.Companion.2012.76.

[2] P. Qin, B. et.al "Bandwidth-Aware Scheduling with SDN in Hadoop: A New Trend for Big Data" Proceedings CoRR, 2014.

[3] Nathan Farrington et.al Helios: a hybrid electrical/optical switch architecture for modular data centers, Proceedings of the ACM SIGCOMM 2010 conference

[4] G. Wang, T. E. Ng, and A. Shaikh, "Programming your network at run-time for big data applications," in Proceedings of the first workshop on Hot topics in software defined networks (HotSDN '12). ACM Press, 2012, p. 103.

[5] P. Samadi, V. Gupta, B. Birand, H. Wang, R. Jensen, G. Zussman and K. Bergman "Software-Addressable Optical Accelerators for Data-Intensive Applications in Cluster-Computing Platforms" Journal of Lightwave Technology, vol. 31, pp. 587-593, Feb., 2013